Server selection on the internet using passive probing

José Afonso and Vasco Freitas

Universidade do Minho, Departamento de Informática
P-4709 Braga, Portugal
{mesjaa,vf}@di.uminho.pt

ABSTRACT

This paper describes a server selection mechanism for connection oriented services based on passive probing. The criterion of selection is the quality of service expected from each server, expressed as a function of availability and response time. Measures from previous connections to servers made by local clients are used to continuously update a QoS database which the prediction algorithm uses to compute the response time expected in subsequent connections. The forecasting approach is mainly based on prior measurements of TCP connection establishment time. The maximum segment size (MSS) in a connection is also considered. The proposed metric is compared with other ones normally used to measure network proximity. Results show that the proposed server selection mechanism achieves a reduction of response time of over 50% compared with a random selection mechanism.

1 INTRODUCTION

In a distributed information system like the World-Wide Web, the use of a single server to supply a given service (or document) may cause problems in terms of quality of service (QoS) offered to the clients. One of the problems is the long delays experienced by clients located far away from the server, as packets have to traverse many networks. Another problem is the load that a popular server is submitted to when flooded by requests from a vast audience, which can increase its access latency even for nearby clients. A solution to these problems is to place replicated copies of documents at other servers spread along the Internet. This technique distributes the load over both the servers and the network and is likely to achieve reduced response times. On the World-Wide Web, a list of servers hosting replicated copies of the same resource, identified by its unique URN (Uniform Resource Name),[2] can be obtained from its URC (Universal Resource Characteristics).[1] However, given a list of servers containing copies of some document a question that arises is which server should a client choose in order to obtain the best quality of service.

In this paper we are concerned with two factors that affect the quality of service from a client's point of view: response time and server availability. The choice of a server should be based upon an estimation of these two parameters, taking into consideration prior measurements for each elegible server. Some probe measurements that can be used to predict these parameters are discussed and their results compared. Response time is defined as the elapsed time between the client sending a connection request to the server and it receiving the data requested. It is the sum of the connection time (meaning connection establishment time) and the transfer time (meaning data transfer time). The response times of a typical server on the Internet present a high variability, as shown on figure 1, where, for several days, the same document was requested using HTTP[3] every hour. In fact, the transfer

rate may change drastically even during a connection. Figure 2 shows four line segments plotting transfer times as a function of bytes transferred for four consecutive network connections in this experience. While in connections 1 and 3 the data transfer rate remains almost constant, large rate variations are observed in connections 2 and 4. Such a behaviour can be explained by sudden variations in network load along the path between the client and

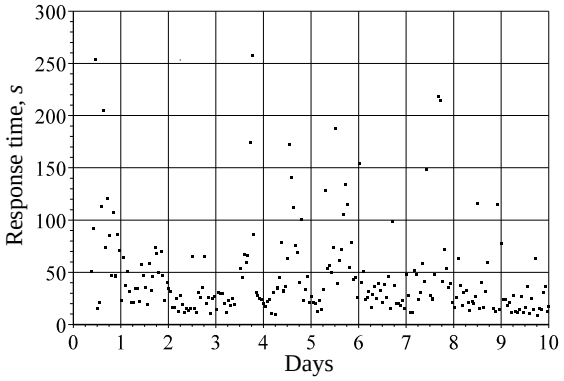

Figure 1: Response times of a server

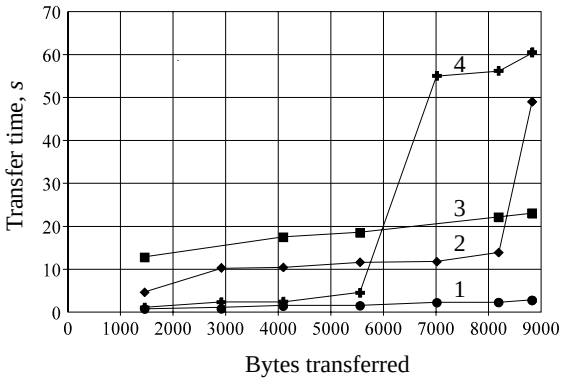

Figure 2: Transfer times of a server

the server caused by competing traffic which is almost impossible to predict with a high degree of precision. As a consequence, the server selection mechanism will be unable to make the right choice all the time, regardless of the method used to make the prediction. Despite the high variability of measurements, they showed that some servers consistently gave better average response times than others. So we took as our main objective to define a forecasting method of server selection that would reduce the average response time required to retrieve a set of documents and compare the results with those obtained by a random selection criterion. Furthermore, the method should:

1. minimize any extra load into the network should probing be necessary, and
2. keep the time overhead of the selection process to the lowest value possible.

2 PREDICTION METHODS AND RELATED WORK

To estimate the proximity of Internet hosts, two different approaches are commonly used. One is based on the distance between hosts and another is based on time measurements. A distance can be interpreted either as geographical or network distance, the later being normally expressed in terms of the number of network hops between hosts. On the other hand, the most used time measurement of proximity is the round-trip latency. The round-trip time (measured using the ICMP echo request and reply messages), is also called ping[15] time, due to the utility commonly used for such measurements. In [4], the authors suggest that the distance in hops between two hosts is not a good estimate of response time. They show that the number of hops between two Internet hosts is not strongly correlated to round-trip latency (they found a correlation of just 10% on their data). Besides, in a simulation of server selection using different metrics, they obtained better results (smaller response times) for the metrics based on measurements of round-trip time than for the metrics based on the number of hops or geographical distance. They also considered the use of two other probe tools in Wide-Area Networks, along with the round-trip time: BPROBE, which estimates the base bandwidth of a connection and CPROBE, which estimates the current congestion of a connection. Both are built using ICMP echo messages.

BPROBE sends multiple packets of different sizes and tries to estimate the bandwidth based on each inter-arrival time and packet size, filtering the results to discard inaccurate measurements. CPROBE sends a stream of packets and measures the time between the receipt of the first packet and the receipt of the last packet. Their association of the average round-trip times and bandwidth components was called $Predicted\_TT$ (predicted

transfer time), and is computed as follows:

$$Predicted\_TT = k_1\, RTT \;+\; k_2\, \frac{S_{doc}}{B_{avail}}$$

Where RTT is the average round-trip time, S_{doc} is the document size and B_{avail} is the available bandwidth calculated using BPROBE and CPROBE. The regression analysis correlation shows an improvement in accuracy of transfer time estimation made with $Predicted\_TT$ over the estimation based on a single ping, and suggests that a single ping (used by some authors[5] as prediction metric) is not a good predictor of response time. Further measurements showed that the average of five round-trip times gives a result very similar to that of $Predicted\_TT$, so that it is concluded that the average of round-trip times alone already accounts for congestion effects. It seems that the use of separate probes for bandwidth and congestion would have a high cost in terms of network load for a small increase in prediction accuracy.

3 PASSIVE PROBING

Our analysis was made for HTTP, even though it can easily be extended to other application level protocols that rely on TCP,[6] like FTP. Based on the results of other authors,[5] we decided to use a time criterion, with the average of a set of measurements as the main variable to estimate the response time. The most referred to time metric for the measurement of network proximity is the round-trip time based on the ICMP echo messages (ping). We are mainly concerned with server selection for services that use connection oriented protocols. Therefore we also considered two other metrics:

a) the time required to establish a connection (which we call connection time) and
b) a function of the response time and the number of bytes transferred.

As opposed to the ping metric, these two allow the use of passive probing. The main advantage of this technique is that there is no need to load the network with more traffic in order to make the measurements. The results of prior connections made by local clients to a server can be used to predict the response time for the next connection. The use of passive probing to measure the quality of service expected from servers has been considered before by others. In [7], the author uses the passive probing concept on the X.500 Directory to provide quality of service information to the users, while in [8] they consider it to make URL[9] selection.

3.1 Collection of measurements

Measurements collected by passive probing should be stored in a QoS table for further use. The use of one table per client, with data gathered from its own accesses only is not satisfactory. A better configuration consists of a single QoS table shared by all local clients, with measurements collected from all of them. The larger the number of clients, the larger the number of connections registered and the more complete and up to date the data in the table will be. If clients use a proxy server,[12] their requests are made by that proxy, which could itself collect the measurements from its own accesses and also host the QoS table. The selection of a server to supply a document requested by any client should be easier for the proxy to make as the data required for the decision is readily available. However, management of the QoS table should be kept independent of proxies and even clients. To accomplish this, network traffic is monitored and selected packets are captured. IP and TCP packet headers and, if necessary, parts of the application level protocol header contain most of the information needed for making measurements, at least more than what the clients themselves would normally provide.

We use *tcpdump*[10] to capture network traffic. *tcpdump* provides for packet filtering and dumping of filtered packets into a buffer. Since the aim is to look at HTTP connections, only TCP packets that use port 80 are filtered in at which time they receive a time stamp. Not all of these packets are needed though. SYN packets identify connection initiation time and FIN packets allow computing the total time of HTTP connections as well

as the number of bytes transferred. A filter, which takes the form of a boolean expression (see [10]), specifying exactly such packets is given as an argument to *tcpdump*. The data needed for each of the two metrics proposed before is thence readily available.

3.2 Comparison among metrics

The metric based on response time and transferred bytes (metric b in section 3) was excluded because of the many obstacles it presents to its implementation compared to the metric connection time, as explained below.

1. Many connections would have to be discarded, considerably reducing the data from the servers available to update the QoS table:
   - HTTP connections with POST method should be discarded because it could imply additional processing time on server. The same could happen to method GET directed to a cgi script.
   - HTTP connections with method GET and absolute-path URL should be discarded because the destination of the request is a proxy server, but the response time will depend on the server referenced by the URL.
   - To obtain good precision on the measurement, a large number of bytes transferred would be desirable because the data is transferred in packets. However a considerable number of connections examined showed only a few bytes coming from server, such as those with HTTP responses of type 'not modified'.
2. The program that collects the data from connections become complex and protocol dependent
   - It is necessary to dump the PUSH packets of client and examine the HTTP method to select the connections.
   - RESET packets should be dumped so that connections canceled by either the client or the server during the transfer of data are detected.
   - The use of persistent HTTP connections[11] would make difficult the identification of the end of a request and the beginning of the next.
3. A function relating the response time and the bytes transferred that would apply to all conditions would have to be found.

A better approach that avoids all the problems mentioned is to simply use the connection time (metric a in section 3), with the following advantages:

1. It is independent from the document requested from a server.
2. All connections may be considered in the QoS table.
3. There is no need to examine the HTTP header.
4. It can be easily extended to other protocols, like FTP.
5. Connections are simpler to monitor.

4 BEHAVIOUR OF CONNECTION TIMES

Like response times, connection times also show a high variability. Figure 3 plots the connection times measured in the experiment reported by figure 1 and figure 4 shows its frequency distribution. Values below 2.5 s correspond to the round-trip time to the server and are similar to the ping times. Both times are affected by delays on the network path caused by competing traffic. Connection times take also into account the server load and its availability, while in ping replies are issued by a different process on the host and do not reach the server process. Another difference between ping and connection times is that the later presents various regions of concentrated

values above 2.5 *s*. These regions correspond to retransmissions due to protocol timeouts in both the client and the server. Retransmission are mainly caused by packets being lost due to network congestion. With ping, a lost packet would mean absence of response from the server. The occurrence of retransmissions can be very useful

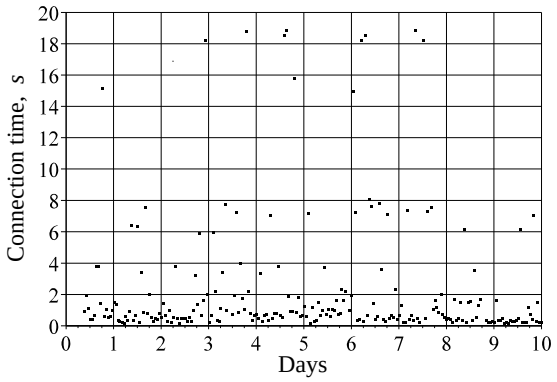

Figure 3: Connection times of a server

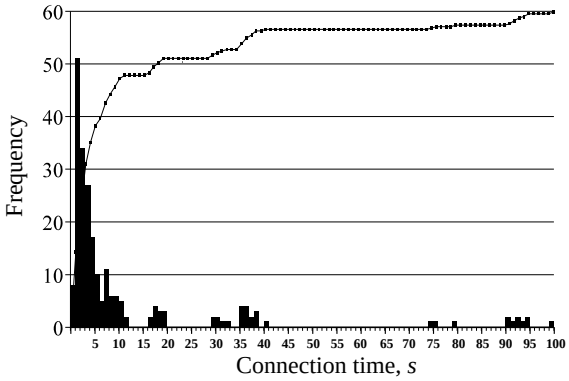

Figure 4: Distribution of connection times

in predicting response times. Among servers that give similar values of round-trip times, the frequency of lost packets can make the difference in terms of response time. The frequency distribution of connection times (and round-trip times as well), shows a non-linear behavior typical of exponential or gamma distributions, where the majority of occurrences are contained below the mean. As a consequence, the median is much smaller than the average. Our measurements show that a geometric mean produces better forecasting results than an arithmetic mean, specially when the number of samples available is small. The geometric mean can be computed by taking the anti-logarithm of the arithmetic mean of the sample's logarithms.

5 EXPERIMENTAL RESULTS

In order to compare metrics that could be used to estimate response times, a HTTP client was modified to record connection and response times. The client was used on two local hosts, each with its Maximum Segment Size (MSS) option set to a different value. The MSS specifies the maximum sized segment that the sender wants to receive and it is the largest block of data that TCP will send. For each host, a set of 14 HTTP servers on the Internet containing the same document (14 kbytes) was accessed during a period of 10 days with 1 hour interval between requests. The host for the first set of measurements was a Soulbourne (SunOS 4.1.3), with MSS of 536, the default value, and the host for the second set was a SUN-server10 (SunOS 5.3) with MSS of 1460, the maximum allowed by a true Ethernet encapsulation.

5.1 First set of measurements

The results for the first host, ordered by the arithmetic mean of response times, are shown in table 1. Each server is identified by its country. The client was located in Portugal. Time values shown are averages of all measurements done for each server, where A.M. is the arithmetic mean and G.M. is the geometric mean. Transfer fails correspond to the sum of connection establishment failures and failures due to timeouts during data transfer (5 minutes were allowed). Round-trip times only consider connection times that took less than 2.5 *s*. The number of network hops was measured with traceroute.[14] The MSS announced by the server does not influence the results in this case. Its usefulness will be explained later. The correlation between response times and the corresponding connection times for all servers considered together was just 25%, showing that a single measurement of connection time provides a poor forecast even for its own response time. On the other hand, the average of response times

and connection times show a much higher correlation, i.e., 95% for G.M. and 93% for A.M., also higher than the correlation between average response times and round-trip times which were calculated at 75% and 56% respectively.

Table 1: First set of measurements

Country of Server	Resp time AM	Transfer fails (%)	Connection time		Round-trip time		Number of hops	Server MSS
			GM	AM	GM	AM		
Netherlands	32.60	0.4	0.53	1.12	0.41	0.57	12	1460
Germany	36.22	4.2	0.55	0.95	0.46	0.59	16	1460
Italy	36.88	10.6	0.66	1.38	0.50	0.63	12	1460
Spain	42.86	5.0	0.81	3.00	0.36	0.47	14	1460
United Kindom	51.46	3.9	1.00	3.89	0.34	0.39	16	536
France	52.23	1.5	1.09	2.84	0.63	0.80	18	1460
Norway	62.86	13.4	1.14	3.29	0.45	0.53	14	512
Finland	64.35	5.0	0.98	3.85	0.35	0.43	11	512
U.S.A (4)	70.00	2.1	1.56	3.72	0.69	0.80	20	1460
U.S.A (2)	94.37	6.8	1.73	4.99	0.63	0.70	19	1460
U.S.A (3)	101.45	15.3	1.74	5.16	0.65	0.70	20	1460
Austria	101.64	10.0	1.42	4.19	0.58	0.65	14	536
U.S.A. (1)	110.98	12.4	1.69	5.17	0.64	0.69	19	1460
Czech Rep.	138.02	11.4	2.46	7.17	0.78	0.86	14	512
Correlation with response time			0.95	0.93	0.75	0.56	0.34	

Another important result is that the order of selection based on average connection times would be much closer to the order of average response times than to the order of average round-trip times. In this case, the sequence for geometric mean of connection times matches the sequence for average response times for the first four servers. On the other hand, the first server selected considering round-trip averages would be the fifth on the average response time sequence. The correlation between response time and number of hops is only 34% and the nearest server in terms of network hops is ranked eighth on the list of average response times. This confirms the number of hops as a bad predictor. The optimal average response time for this set of measurements is 21.24 *s*. It is only 35% lower than the value of 32.60 *s* obtained for the server from Netherlands. On the other hand, the worst average response time is 147.65 *s* (4.35 times higher), and the rate of transfer failures is 50% (125 times higher). The average response time of all servers is 71.17 *s* and the average rate of transfer failures is 7.3%. Considering these values for a case of random selection, the use of the Netherlands server over all the period of measurement provides a reduction of 54% in response time and 95% in transfer fails. These results suggest that even if the same server is used for a long period of time, the reduction in response time can be considerable. The results in table 1 also indicate that round-trip times produce erroneous forecasts if lost packets are not considered.

5.2 The influence of MSS

Our measurements show that the maximum segment size (MSS) of a connection influences transfer time. The MSS to be used by both the client and the server are defined at the beginning of a connection when SYN packets are exchanged. The smallest value is used by both sides. When the MSS is not announced, a default value of 536 is adopted. Usually, each host has a defined MSS used by all clients and servers on that host. The MSS to be used on a connection can be predicted given the MSS of each party. The server's MSS can be stored on the QoS table. The client's MSS can be supplied along with the query or obtained from a table relating local client's IP address and their respective MSS. A higher MSS means that more data is transferred on a single packet. The TCP slow-start mechanism limits the number of packets that can be transferred without acknowledgment at the

beginning of a connection. On a congested network the limitation persists. In such cases, the use of a higher MSS can significantly reduce transfer time, in the absence of IP fragmentation along the path. The client's MSS used in table 1 is 536. This value is used in all connections, except when the server has a MSS of 512 where this value is used instead. These two values are very close so the influence of MSS is minimal.

5.3 Second set of measurements

Another set of measurements with the same characteristics of those in table 1 was collected with another host having a MSS of 1460. In this case, the MSS used in connections was the proper server's MSS, shown in table 1. The results in terms of correlation are presented in table 2. The first row of data shows the correlations between

Table 2: Second set of measurements

Correlation with response time	Connection time		Round-trip time	
	GM	AM	GM	AM
Considering measured times only	0.50	0.65	0.23	0.06
Considering also MSS of connections	0.79	0.89	0.77	0.73

the averages of response times and connection and round-trip times. Values are lower than those found in table 1. In this set of measurements, ordering based upon average response times also changed. Connections with MSS of 1460 took precedence over the others. Comparing the average response times in the two sets of measurements we found that, on the average, response time is approximately inversely proportional to the square root of the connection's MSS. We weighted the average times with the respective connection's MSS to obtain new prediction values and computed the correlation between these values and the average response times. The second row in table 2 shows improved results compared to those on the first row confirming that the influence of MSS is indeed quite relevant.

6 AVAILABILITY

Another disadvantage of ping is that it does not measure server availability, only host availability. A host can be alive and responding to ICMP echo requests, but the process responsible for a service (HTTP, in this case), may be inactive. Such situation was observed during our measurements. Using connection time as the metric, two methods are considered to estimate the availability of a server. Long term availability can be obtained by dividing the number of successful responses by the number of connection requests made by local clients over the period of measurement. Both the number of requests and the number of responses are stored in the QoS table for each server. The current availability estimation takes into account the last connection trials. Each failed connection reduces the value of the estimation. When a connection succeeds the current availability estimation is set to its maximum value. The current availability estimation for each server is directly stored in the QoS table.

7 A QoS SERVICE

One possible way to make the data in the QoS table available to clients is to provide it as a separate service accessible using a protocol, as proposed in [5]. With the passive probing technique, servers are not probed during the processing of a query, so the response is given immediately. In our case, the prediction value for each server is computed from the average connection time, the connection's MSS and the estimated availability. With a local

host providing the QoS service, the cost of making a query is just the exchange of a pair of UDP datagrams (a few milliseconds). The benefit is a considerable reduction in the time required to retrieve a document. Some servers may have no information available in the QoS table. This is likely to happen for distant or unknown servers, which might turn out to be bad options for selection after all.

In [5] and [4], the authors assume that a list of IP addresses should be supplied to the server selection process. In many cases, however, a client would first obtain the servers' domain names and then access the DNS to resolve each of those names. This procedure presents two drawbacks:

1. it loads the network with many DNS queries but only one will be selected.
2. delay is increased as the client has to wait for all DNS replies to obtain the full list of IP addresses.

We propose that the service responsible for server selection should be able to handle not only IP addresses but also domain names without querying the DNS. To achieve this, both domain names and IP addresses collected by passive probing are stored in the QoS database. However, HTTP packets only contain the server's IP address, not the domain name. *Tcpdump* provides an option to convert IP addresses to domain names, but it has some problems:

- *tcpdump* returns the host's canonical name and the name by which the server is known is usually an alias;
- *tcpdump* uses the DNS to obtain the server's name;
- sometimes the domain name resolution fails.

A solution to these problems is obtained by having *tcpdump* to also monitor DNS traffic. Before contacting a server, clients usually query the DNS to obtain its IP address. By comparing the IP addresses contained in both the HTTP packets and the DNS response packets it is possible to recover the domain name used by a client. As the QoS service can supply the server's IP address given its domain name, clients may bypass the usual query to the DNS.

8 IMPLEMENTATION

Traffic is monitored on the external access of the University campus network by a SunSparc station under SunOS 4.1.2.2. running *tcpdump*. HTTP and DNS packets are passed to a program that gathers and computes the relevant data and records it in a QoS database. This procedure runs in real time and therefore the database is updated as soon as a connection ends. During a period of three months, the number of HTTP servers registered in the database was more than 25000.

8.1 The QoS database

The QoS database was implemented using hash tables for fast data retrieval. The main table is indexed by server IP address and contains the following fields:

- The server's IP address.
- The expected connection time.
- The server's recent availability estimation.
- The server's MSS.
- The number of connection requests made by local clients during the probing period.
- The number of successful connection confirmations during the same period.
- The time of the last connection.

- A pointer to a list of domain names associated to this server.

Another hash table indexed by domain names of servers contains the following fields:

- A domain name.
- A pointer to the associated server record on the main table.
- A timestamp for the last time the name was detected on a connection.

The expected connection time is computed from the single exponential smoothing forecast[13]

$$F_n \;=\; \alpha\, X_n \;+\; (1-\alpha)\, F_{n-1}$$

where F_n is the current forecast value, X_n is the current measured value, F_{n-1} is the previous forecast value and α is a real constant in the interval [0,1], whose value is chosen according to the desired influence of the new measured value on the forecast. A property of single exponential smoothing is that the weight of previous samples decreases exponentially (when α is a constant). The geometric mean of the samples is used instead of the arithmetic mean, so, the logarithm of the measured connection time is used for X_n, and the forecast is obtained from the inverse logarithm of F_n.

9 NETWORK LOAD

Network load, which considerably influences response time, varies with the time of the day. Figure 5 shows a moving average of the response times plotted in figure 1. The first day on the chart (day 1) is a Friday. Expected

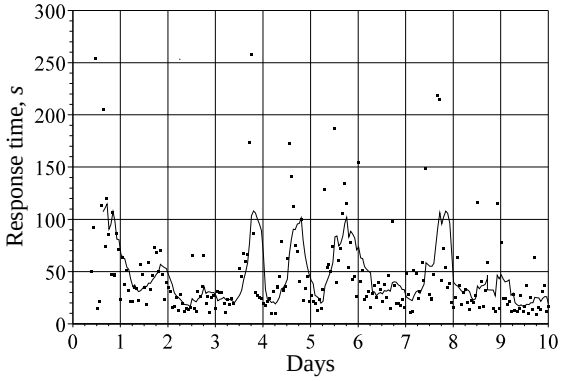

Figure 5: Average response times of a server

response time is seasonal. It follows a regular oscillatory pattern peaking very high between late morning and early afternoon and quite low from late in the evening until early in the morning on weekdays (days 4, 5, 6 and 8). Its values and hourly variation are much reduced on weekends (days 2, 3, 9 and 10) and bank holidays (day 7). Connection times are also influenced by network load. As servers are probed at different times of the day, that is, whenever clients are accessing them, response time forecasts must have a similiar seasonal behaviour. In our implementation connection time forecasts are compensated for expected network load which is estimated from data collected for all servers.

10 CONCLUSIONS

The approach described is simple, nevertheless it gives useful results. The use of a time criterion to estimate the response time gives much better results than the static criterion of number of hops. However, a single

measurement is not sufficient to predict the response time, due to their high variability. The use of an average of measurements results much better.

Server selection based on connection times gave better results than selection based on round-trip times. Network load, server load and availability are better accounted for when connection times are used. The MSS also showed to influence results. An advantage of passive probing is that it does not add any extra load onto the network. On the other hand, there is no control over how often and when measurements are taken to each server. The data stored in the QoS table can be useful for other purposes besides selection of servers. Information on the most accessed servers by local clients, on servers with best and worst response times, on the distribution of accesses by domains and other statistics, can be obtained from the QoS database.

11 REFERENCES

[1] Daniel Jr., R. and Mealling, M., *URC Scenarios and Requirements*, Internet Draft draft-ietf-uri-urc-req-00.txt, work in progress from the IETF, Nov 1994.

[2] Hoffman, P. and Daniel Jr., R. *Uniform Resource Names (URNs)*, Internet Draft draft-ietf-uri-yaurn-00.txt, work in progress from the IETF, Mar 1995.

[3] Berners-Lee, T. *et al*, *Hypertext Transfer Protocol - HTTP/1.0*, RFC 1945, IETF Network WG, May 1995.

[4] Carter R., Crovella, M., *Dynamic Server Selection using Bandwidth Probing in Wide-Area Networks* Tech Rep BU-CS-06-007, Computer Science Department, Boston U, Boston MA 02215, USA, Mar 1996.

[5] Cox, J., Green, S., Moore, K., *SONAR – A Network Proximity Service*, Internet Draft draft-moore-sonar-01.txt, work in progress from the IETF Network WG, Feb 1996.

[6] Postel, J. (ed.), *Transmission Control Protocol - DARPA Internet Program Protocol Specification*, RFC 793, Information Sciences Institute, U of Southern California, Marina del Rey, CA 90291, USA, Sep 1981.

[7] Barker, P., *Providing the X.500 Directory User with QoS Information*, Computer Communication Review, ACM SIGCOMM'94, pp 28-37, 1994.

[8] Rio, M., Macedo, J., Costa, A., Freitas, V., *Supporting a URI infrastructure by Message Broadcasting*, Hyper-Media Proc INET'95 - 1995 ISOC Int Networking Conf, http://info.isoc.org/HMP/PAPER/116/ps/paper.ps, Honolulu, Hawaii, USA, Jun 27-30, 1995.

[9] Berners-Lee (CERN), *et al*, *Uniform Resource Locators (URL)*, IETF Network WG, RFC 1738, Dec 1994.

[10] McCanne, L, Leres, C and Jacobson, V., *Tcpdump software*, Lawrence Berkeley National Laboratory Network Research Group, tcpdump@ee.lbl.gov, 1996. Available from ftp://ftp.ee.lbl.gov/tcpdump.tar.Z.

[11] Mogul, J., *The Case for Persistent-Connection HTTP*, Digital Equipment Corporation Western Research Laboratory, 250 University Ave, Palo Alto, CA 94301, *In* ACM SIGCOMM'95, Cambridge MA USA, 1995.

[12] Luotonen, A. and Altis, K., *World-Wide Web Proxies*, *In* Computer Networks and ISDN Systems 27(2): 147-154, Apr 1994.

[13] Makidrakis, S. *et al*, *Forecasting methods and applications*, John Wiley & Sons, pp 84–111, 1983.

[14] Van Jacobson, *Traceroute software*, Lawrence Berkeley National Laboratory Network Research Group, traceroute@ee.lbl.gov, Dec 88. Available from ftp://ftp.ee.lbl.gov/pub/traceroute.tar.Z.

[15] Michael, Muuss, *Ping software*, U. S. Army Ballistic Research Laboratory, Dec 1983. Available from ftp://uunet.uu.net/bsd-sources/src/ping.